APLIKASI *PATTERN MATCHING* UNTUK VALIDASI PESAN PADA *GAME ONLINE* DAN *SOCIAL NETWORK*

Wishnu / 13511040
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung Jl.Ganesha 10 Bandung 40132, Indonesia
cloud_strife1310@yahoo.com

Abstrak – Di era digital dan serba modern ini, manusia hampir tidak dapat dipisahkan dari internet. Internet digunakan untuk berbagai macam hal, mulai dari pencarian informasi, melakukan transaksi jual beli, penggunaan jejaring sosial (social network), hingga untuk sekedar bermain game online. Khususnya untuk jejaring sosial dan game online, kita akan berinteraksi dengan pengguna yang lain secara real-time / tepat pada waktu itu juga. Makalah ini akan membahas tentang bagaimana validasi terhadap pesan negatif yang dikirimkan sesama pengguna agar tidak mengganggu komunikasi antar sesama pengguna.

Kata Kunci – Game Online, Jejaring Sosial, Pesan, Pattern Matching.

I. PENDAHULUAN

Media sosial adalah sebuah media online, yang dimana para penggunannya bisa dengan mudah berpartisipasi, berbagi, dan menciptakan isi meliputi blog, jejaring sosial, pengetahuan, forum, dan dunia virtual. Blog, jejaring sosial dan pengetahuan merupakan bentuk media sosial yang paling umum digunakan oleh masyarakat di seluruh dunia.

Menurut Kaplan dan Haenlein, media sosial dapat digolongkan ke dalam 6 jenis, yaitu (Proyek Kolaborasi, Blog dan Microblog, Konten, Situs Jejaring Sosial, *Virtual Game World*, dan *Virtual Social World*). Contoh dari keenam jenis media sosial tersebut masing-masing adalah Wikipedia, Twitter, Youtube, Facebook, Game Online, dan Second Life. Pada makalah ini, akan dibahas contoh mengenai jejaring sosial dan *virtual game world*.

Media sosial memiliki ciri-ciri bahwa pesan yang disampaikan tidak hanya satu orang saja, namun bisa ke berbagai orang layaknya SMS. Pesan yang disampaikan juga bersifat bebas dan memiliki waktu penyampaian yang cenderung lebih cepat dibanding media lainnya. Ciri berikutnya adalah bahwa penerima pesanlah yang menentukan waktu interaksi.

Game Online atau yang sering disebut *Online Games* (Permainan Daring) adalah sebuah permainan yang dimainkan di dalam suatu jaringan komputer. Jaringan (*Network*) yang digunakan mulai dari LAN (*Local Area Network*) hingga koneksi Internet. Perkembangan game online tidak lepas dari perkembangan dari jaringan komputer yang sangat pesat dan teknologi multimedia yang digunakan.

Koneksi Internet memungkinkan pemain yang memainkan game online dari seluruh penjuru dunia dapat bertemu di dunia maya, berkomunikasi, bertukar pesan, dan memainkan game online tersebut dalam waktu yang bersamaan. Game online dapat dimainkan dengan komputer, konsol (*console*) seperti PS3 (*Play-Station 3*), dan Xbox360 yang memiliki koneksi internet.

Pada makalah ini, penulis akan membahas tentang penggunaan pattern matching untuk validasi pesan yang kurang baik / negatif yang dikirimkan pengguna saat memainkan game online maupun menggunakan jejaring sosial. Langkah yang dapat diambil setelah validasi dilakukan dapat berupa mengganti kata-kata yang kurang baik / negatif di dalam pesan dengan kata-kata yang baik maupun memberikan peringatan kepada pengguna bahwa pesan yang dikirimkan mengandung kata-kata yang kurang baik / negatif.

II. *PATTERN MATCHING*

Pattern Matching atau sering disebut juga pencocokan *string* adalah suatu cara/algoritma untuk melakukan pencarian semua kemunculan *string* pendek (*pattern*) di dalam *string* yang lebih panjang (*teks*). Untuk memperjelas maksud dari *pattern matching* ini, akan diberikan sebuah ilustrasi sebagai berikut :

1. T : Teks (*text*), yaitu (*long*) *string* yang panjangnya *n* karakter.
2. P : *Pattern*, yaitu *string* dengan panjang *m* karakter (asumsi $m <<< n$) yang akan dicari di dalam teks.

Kemudian, carilah (*find* atau *locate*) lokasi pertama di dalam teks yang bersesuaian dengan *pattern*.

Contoh :
T : “the rain in spain stays mainly on the plain”
P : “main”

Dengan menggunakan algoritma *pattern matching*, *pattern* “main” akan ditemukan tepat setelah kata ”*stays*” yaitu pada kata “*mainly*”.

Algoritma *pattern matching* ini memiliki banyak sekali pengaplikasiannya dalam ilmu pengetahuan. Contohnya adalah pencarian di dalam editor text, *web search engine* (misal : Google, Yahoo, dll), analisis citra, hingga ke aspek bioinformatika (misal : pencocokan rantai asam amino pada rantai DNA, penentuan senyawa hasil reaksi kimia, dll).

Sebuah *string* dapat kita pandang memiliki prefix dan suffix (awalan dan akhiran) yang dapat menjadi ide dasar dalam algoritma *pattern matching*. Asumsi S adalah sebuah *string* dengan panjang m.

$$S = x_1x_2...x_m$$

Prefix dari S adalah sebuah *substring* S[1...k-1] .
Suffix dari S adalah sebuah *substring* S[k-1...m] dengan :

- k adalah sebuah indeks di antara 1 dan m
- S[0] adalah sebuah karakter *null*, dan bersimbol ∅

Contoh :

S :

a	n	D	r	e	w

Semua kemungkinan prefix dari S adalah :
- “∅”, “a", "an", "and", "andr”, "andre“.

Semua kemungkinan suffix dari S adalah :
- “∅”, “w", “ew", “rew", “drew", “ndrew”.

Di bawah ini adalah beberapa contoh algoritma *pattern matching*. Mereka adalah :

A. Algoritma *Brute Force*

Algoritma *Brute Force* merupakan algoritma pencocokan *string* yang ditulis tanpa memikirkan peningkatan performa. Algoritma ini mengecek setiap posisi di dalam *text* T untuk melihat apakah sebuah *pattern* P dimulai pada posisi tersebut. *Pattern* P bergeser satu karakter pada setiap waktu melalui T.

Secara sistematis, langkah-langkah yang dilakukan algoritma *brute force* pada saat mencocokkan string adalah:

1. Algoritma *Brute Force* mulai mencocokkan *pattern* pada awal teks.
2. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
    1. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*mismatch*).
    2. Semua karakter di *pattern* cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Algoritma kemudian terus menggeser *pattern* sebesar satu ke kanan, dan mengulangi langkah ke-2 sampai *pattern* berada di ujung teks.

Algoritma *Brute Force* memiliki kompleksitas waktu rata-rata O(m+n). Algoritma ini akan cepat apabila karakter di dalam sebuah teks bernilai besar (misal : A..Z, a..z, 1...9, dll). Namun, algoritma ini akan lambat apabila karakter di dalam sebuah teksa bernilai kecil (misal : 0,1 (data biner)).

B. Algoritma Knuth-Morris-Pratt (KMP)

Algoritma Knuth-Morris-Pratt adalah salah satu algoritma pencarian *string* yang dikembangkan secara terpisah oleh Donald E. Knuth dan James H. Morris bersama Vaughan R. Pratt.

Algoritma KMP melihat *pattern* di dalam teks dari kiri ke kanan layaknya algoritma *brute force*. Namun, algoritma ini menggeser *pattern* secara ”lebih cerdas” dari pada algoritma *brute force*.

Algoritma KMP menggunakan fungsi pinggiran KMP (KMP *Border Function*) sebagai praproses *pattern* untuk mencari kecocokan dari prefix dari *pattern* dengan *pattern*nya sendiri.

- j = posisi yang tidak sama di dalam P[]
- k = posisi sebelum ketidakcocokan (k = j-1)
- B*order function* b(k) didefiniskan sebagai panjang dari prefix yang paling panjang dari P[1..k] yang juga merupakan suffix dari P[1..k]
- Nama lainnya : *failure function* (disingkat: *fail*)

Contoh *Border Function* :

P : “abaaba”
j : 123456

j	1	2	3	4	5	6
P[*j*]	a	b	a	a	b	a
b(*j*)	0	0	1	1	2	3

Tabel 1 : Contoh KMP *Border Function*

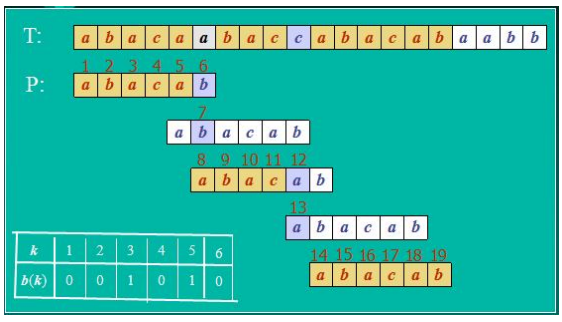
b(j) adalah ukuran dari border yang paling besar. Di dalam kode program, b() direpresentasikan dengan array, seperti yang di tabel.

Secara sistematis, langkah-langkah yang dilakukan algoritma Knuth-Morris-Pratt pada saat mencocokkan *string* adalah :

1. Algoritma Knuth-Morris-Pratt mulai mencocokkan *pattern* pada awal teks.
2. Dari kiri ke kanan, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
    1. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*mismatch*).
    2. Semua karakter di *pattern* cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Algoritma kemudian menggeser pattern berdasarkan tabel next, lalu mengulangi langkah 2 sampai *pattern* berada di ujung teks.

Kompleksitas waktu algoritma KMP adalah O(m+n). Algoritma KMP tidak akan pernah bergeser mundur pada teks. Namun algoritma KMP tidak cocok pada *pattern* dan teks dengan ukuran yang besar (Alfabet, Numerik,dll).

Ilustrasi cara kerja algoritma KMP digambarkan pada gambar di bawah :



Gambar 1 : Ilustrasi cara kerja algoritma KMP

C. Algoritma Boyer-Moore

Algoritma Boyer-Moore adalah salah satu algoritma pencarian *string* yang dipublikasikan oleh Robert S. Boyer dan J. Strother Moore. Algoritma ini dianggap sebagai algoritma yang paling efisien pada aplikasi umum. Tidak seperti algoritma pencarian *string* sebelumnya, algoritma Boyer-Moore mulai mencocokan karakter dari sebelah kanan *pattern*. Ide dibalik algoritma ini adalah bahwa dengan memulai pencocokan karakter dari kanan, dan bukan dari kiri, maka akan lebih banyak informasi yang didapat.

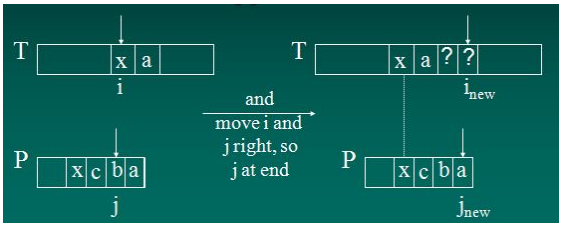
Algoritma Boyer-Moore didasarkan pada 2 teknik, yaitu :

1. *Looking-Glass technique*
- Teknik mencari *pattern* di dalam teks dengan cara bergeser mundur melalui *pattern* yang dimulai dari karakter paling terakhir / belakang

2. *Character-Jump technique*
- Teknik ini dilakukan apabila ketidakcocokan terjadi pada indeks ke-i pada teks dan karakter pada indeks ke-j pada *pattern* itu tidak sama dengan indeks ke-i pada teks.

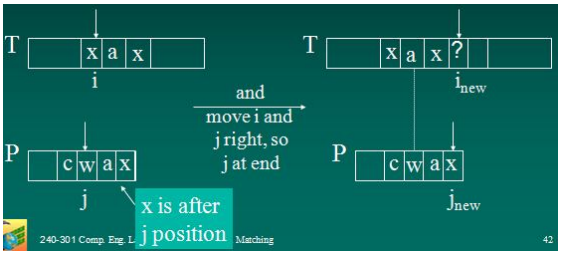
Algoritma Boyer-Moore memiliki 3 kasus yang dapat terjadi di dalamnya, yaitu :

1. Jika *pattern* P memiliki karakter x di dalamnya, lalu akan dicoba untuk menggeser P ke kanan hingga sejajar dengan kemunculan terakhir dari x di dalam P dengan indeks ke-i pada teks.



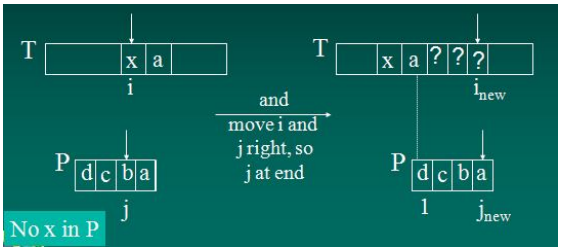
Gambar 2 : Kasus 1 pada Algoritma BM

2. Jika *pattern* P memiliki karakter x di dalamnya, namun apabila penggeseran ke kanan hingga kemunculan terakhir dari x itu tidak memungkinkan, maka geser P ke kanan sebanyak 1 karakter hingga indeks ke-i+1 pada teks.



Gambar 3 : Kasus 2 pada Algoritma BM

3. Apabila kasus 1 dan 2 tidak dapat diterapkan, lalu geser *pattern* P hingga indeks pertama pada *pattern* sejajar dengan indeks ke-i+1 pada teks.



Gambar 4 : Kasus 3 pada Algoritma BM

Algoritma Boyer-Moore memiliki praproses terhadap *pattern* P dan alfabet A untuk membangun fungsi L() / *last occurrence function*. L() memetakan semua huruf pada A menjadi integer.

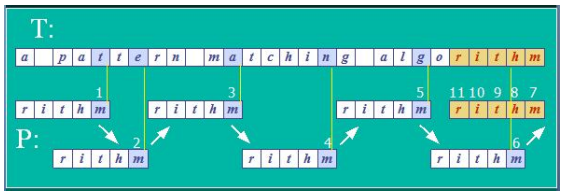
L(x) didefinisikan sebagai indeks i yang paling besar sehingga P[i] == x atau memiliki nilai -1 apabila tidak ada indeks yang didapatkan dengan x adalah sebuah huruf pada A.

Secara sistematis, langkah-langkah yang dilakukan algoritma Boyer-Moore pada saat mencocokkan *string* adalah:

1. Algoritma Boyer-Moore mulai mencocokkan *pattern* pada awal teks.
2. Dari kanan ke kiri, algoritma ini akan mencocokkan karakter per karakter *pattern* dengan karakter di dalam teks yang bersesuaian, sampai salah satu kondisi berikut dipenuhi:
    1. Karakter di *pattern* dan di teks yang dibandingkan tidak cocok (*mismatch*).
    2. Semua karakter di *pattern* cocok. Kemudian algoritma akan memberitahukan penemuan di posisi ini.
3. Algoritma kemudian menggeser *pattern* dengan memaksimalkan nilai penggeseran *good-suffix* dan penggeseran *bad-character*, lalu mengulangi langkah 2 sampai *pattern* berada di ujung teks.

Kompleksitas untuk algoritma Boyer-Moore dapat dinyatakan dalam O(n + x) dengan x adalah besar ruang alfabet. Algoritma ini akan cepat apabila alfabet A memiliki ukuran yang besar, dan lambat apabila ukuran alfabet nya kecil (contoh : cepat untuk pencarian pada teks berbahasa Inggris, lambar untuk pencarian pada teks biner).

Ilustrasi cara kerja algoritma Boyer-Moore digambarkan pada gambar di bawah :



Gambar 5 : Ilustrasi cara kerja algoritma BM

III. VALIDASI PESAN DENGAN MENGGUNAKAN KONSEP *STRING MACTHING*

Melalui media sosial, manusia dapat berinteraksi dengan manusia lainnya di berbagai belahan dunia pada waktu yang sama, atau dengan kata lain secara instan / sangat cepat. Manusia dapat saling berkomunikasi dan berbagi konten melalui media sosial tersebut. Kegiatan yang paling sering dilakukan oleh sesama manusia dalam berinteraksi pada waktu yang bersamaan adalah saling bertukar pesan / mengobrol / *chatting*. Kegiatan mengobrol ini dapat dilakukan di dalam *game online* atau di dalam jejaring sosial apabila 2 atau lebih pengguna *game online* atau jejaring sosial ini terhubung ke internet.

Tidak ada yang dapat menjamin bahwa pesan yang dikirimkan saat masing-masing pengguna tersebut bermain *game online* atau menggunakan jejaring sosial adalah selalu bermakna positif atau netral tanpa adanya kata-kata atau makna negatif di dalamnya. Sampai sekarang hanya sedikit penyedia jasa (*server*) *game online* dan jejaring sosial yang memberlakukan validasi atau pengubahan isi pesan apabila pesan yang dikirimkan masing-masing pengguna mengandung kata-kata atau makna negatif (misal kata-kata kasar, mengandung unsur SARA).

Ide yang mendasari makalah ini adalah bagaimana pesan-pesan negatif yang terkandung di dalam pesan yang dikirimkan pengguna *game online* dan jejaring sosial dapat divalidasi dan diubah agar dapat meminimalisir seminimal mungkin agar makna negatif dan yang tidak baiknya tidak ditampilkan pada pengguna yang menerima pesan negatif/kurang baik tersebut.

Dengan adanya validasi terhadap pesan yang dikirimkan tersebut, fenomena *cyber-bullying* atau yang kita kenal sebagai sejenis penghakiman terhadap seseorang yang dilakukan di dunia maya

(internet) dapat berkurang. Dan jika ingin melakukan kritik dan saran yang mungkin mengandung sedikit makna negatif di dalamnya dapat memakai kata-kata yang lebih halus dalam penyampaiannya.

Ide-ide yang digunakan sebagai validasi pesan yang dikirimkan melalui *game online* dan jejaring sosial meliputi :

1. Menyensor langsung kata-kata negatif di dalam pesan.
2. Mengganti kata-kata negatif di dalam pesan menjadi kata-kata positif yang merupakan lawan kata dari kata-kata negatif atau kata-kata plesetan dari kata-kata negatif tersebut.
3. Memberikan peringatan kepada pengguna yang mengirimkan pesan dengan kata-kata negatif bahwa pesan yang ingin dikirim mengandung kata-kata yang negatif atau kurang baik.

A. Penyensoran Langsung

Pemain *game online* maupun pengguna jejaring sosial dapat mengirimkan pesan ke sesama pengguna lainnya melalui fitur *chatting* maupun menge-*post* sebuah status yang berisikan pemikirannya maupun yang ingin mereka bagikan (*share*). Namun apabila terdapat kata-kata negatif baik itu kata kasar yang frontal atau hal-hal berbau SARA misalnya, penyensoran terhadap kata-kata tersebut dapat langsung dilakukan sebagai langkah untuk meminimalisir hal negatif yang dimaksud pengguna yang mengirimkan pesan ditampilkan secara frontal kepada pengguna yang menerima pesan atau membaca pesan tersebut.

Idenya, pada sisi penyedia jasa atau layanan (*server*) *game online* maupun jejaring sosial didefinisikan daftar kata-kata negatif yang sering digunakan oleh penggunanya dalam berkomunikasi. Pendefinisian kata-kata negatif tersebut dapat bertambah seiring berjalannya waktu atau ditemukannya istilah baru yang ternyata bermakna negatif. Kata-kata negatif yang telah didefinisikan tersebut disimpan dalam bentuk file eksternal pada sisi server. File eksternal ini berguna untuk mengecek / memvalidasi setiap kata dalam pesan yang dikirimkan oleh setiap pemain *game online* / pengguna jejaring sosial.

Setelah pengguna selesai mengetik pesan dan menekan tombol “enter” / “post” / “send”, terdapat jeda beberapa saat sebelum pesan tersebut dapat ditampilkan pada halaman Web atau kotak *chatting* yang dapat dilihat oleh masing-masing pengguna di dalam suatu *conversation.* Jeda beberapa saat inilah yang digunakan untuk mengecek apakah terdapat kata-kata yang bermakna negatif di dalam pesan. Apabila ditemukan kata-kata negatif di dalam pesan yang dikirimkan / diposkan oleh pengguna, kata-kata negatif tersebut akan langsung diubah menjadi karakter lain (misalnya bintang (*)) sebelum pesan tersebut ditampilkan pada halaman Web / kotak pesan *chatting.*

Ilustrasi penyensoran yang dilakukan terhadap kata-kata negatif didalam pesan yang dikirimkan pengguna dapat dilihat dibawah berikut :

Gambar 6 : Contoh pesan dengan kata negatif.

Gambar 7 : Penyensoran terhadap pesan yang mengandung kata negatif

Gambar 6 dan 7 mengambil contoh komunikasi antara dua pengguna pada jejaring sosial Facebook. Gambar 6 mengilustrasikan bahwa pengguna akan mengirimkan sebuah pesan dengan kata negatif “ampas” kepada pengguna lainnya. Sesaat setelah tombol enter ditekan, *server* Facebook akan memvalidasi setiap kata didalam pesan “Ampas lu Daniel!” untuk menemukan kata-kata negatif yang terdapat di dalam pesan tersebut. Setelah ditemukan kata “ampas” yang telah terdefinisi sebagai kata-kata negatif di *server* Facebook, kata “Ampas” akan disensor dan diganti semua karakter nya menjadi karakter bintang (*) sebelum pesan tersebut akan ditampilkan kepada penerima pesan. Dan akhirnya pada kotak pesan *chatting* yang dapat dilihat oleh pengirim dan penerima ditampilkan pesan “ ***** lu Daniel !” yang dimana kata negatif “Ampas” telah disensor seutuhnya.

Situasi pada *game online* terkait pengiriman pesan dengan *chatting* dapat dikatakan hampir sama dengan *chatting* pada jejaring sosial. Oleh karena itu, contoh yang diambil hanyalah salah satu saja, yaitu pada jejaring sosial Facebook.

Bagaimana dengan pesan yang mengandung kata tidak dan diikuti oleh kata negatif yang telah didefisinikan oleh server ? Ide berikutnya adalah saat validasi terhadap kata-kata negatif di dalam pesan dilakukan, setiap kata yang terdapat di dalam pesan akan disimpan secara utuh di dalam sebuah arraylist, dan apabila kata negatif telah ditemukan, maka akan dicek juga apakah kata sebelum kata negatif tersebut adalah kata "tidak" atau bukan. Pengecekan dilakukan dengan memanggil arraylist dengan indeks dikurangi satu dari indeks kata negatif yang ditemukan. Apabila kata tidak ditemukan, kata negatif tersebut tidak akan disensor, dengan berasumsi bahwa setiap kata negatif yang didahului oleh kata tidak sebelumnya akan bermakna netral / positif (tanpa memikirkan secara semantik dari kata tidak yang diikuti oleh kata negatif). Dibawah ini adalah ilustrasi dari konsep di atas :

S : "Ternyata si A memang tidak sebodoh yang saya kira"

....

(validasi berlangsung)

....

R : "Ternyata si A memang tidak sebodoh yang saya kira "

Dengan S adalah pesan yang terlihat pada layar *sender* (pengirim pesan) dan R adalah pesan yang terlihat pada layar *receiver* (penerima pesan). Kata negatif "bodoh" tidak perlu disensor karena terdapat kata "tidak" yang mendahuluinya sehingga makna dari pesan tersebut menjadi netral.

Namun konsep penyensoran ini masih dapat menimbulkan makna negatif yang implisit terhadap penerima pesan. Penerima pesan tahu bahwa pengirim pesan akan mengirimkan pesan yang bermakna negatif, namun hanya tersensor saja dan dapat ditebak kata negatifnya berdasarkan jumlah karakter bintang di dalam pesan. Oleh karena itu, usaha untuk meminimalisir makna negatif dari pesan dengan konsep ini masih dapat dikatakan kurang tinggi. Konsep berikutnya, yaitu penggantian kata negatif dengan lawan kata nya maupun kata lain yang berhubungan akan meningkatkan usaha dalam meminimalisir makna negatif dari pesan yang dikirimkan.

B. Penggantian Kata Negatif Langsung

Konsep penyensoran langsung yang dibahas pada poin A, memberikan kesan bahwa penerima pesan masih dapat menerka kata-kata negatif apa yang ingin dikirimkan oleh pengirim pesan.. Oleh karena itu, ide berikutnya adalah bagaimana meminimalisir kesan untuk menerka kata-kata negatif tersebut.

Langkah untuk meminimalisir kesan tersebut adalah dengan cara mengalihkan makna negatif yang dimaksudkan oleh pengirim pesan. Pengalihan tersebut dapat berupa penggantian langsung terhadap kata-kata negatif tersebut. Kata-kata negatif tersebut dapat diganti dengan lawan katanya maupun kata-kata sejenis namun memiliki makna yang netral / positif.

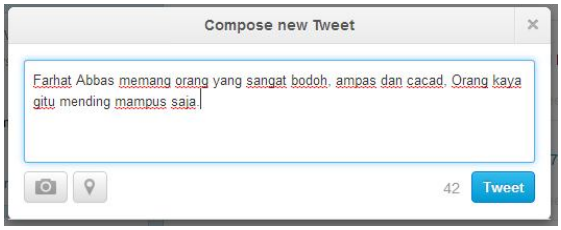
Untuk merealisasikan hal tersebut, data mengenai kata-kata negatif yang telah disimpan harus ditambahkan dengan kata-kata penggantinya. Sebagai contoh, dibawah berikut adalah format dari file eksternal yang telah dimodifikasi dengan ditambahkan kata-kata pengganti kata-kata negatifnya.

```
anjing  : kucing,tikus.
bodoh   : pintar.
ampas   : oke.
cacad   : bagus.
mampus  : hidup.
```

Gambar 8 : Contoh file eksternal yang telah dimodifikasi

Kata-kata disebelah kiri titik dua adalah kata-kata negatif yang telah didefinisikan. Kata-kata disebelah kanan titik dua adalah kata-kata yang akan digunakan untuk mengganti kata-kata negatif tersebut. Apabila pada pesan yang dikirimkan oleh pemain *game online* atau pengguna jejaring sosial ditemukan kata-kata negatif yang sudah terdefinisi pada file eksternal, kata-kata negatif tersebut akan langsung diganti (*replace*) oleh kata-kata pengganti yang telah terdefinisi. Proses penggantian ini dilakukan setelah pengirim pesan menekan tombol enter / *send*. Setelah kata-kata negatif tersebut diganti dengan kata-kata penggantinya, pesan yang sudah tergantikan kata-kata negatifnya akan ditampilkan pada kotak pesan / *chat* yang dapat dilihat oleh pengirim dan penerima pesan. Pesan yang ditampilkan ini sudah tidak mengandung makna negatif lagi.

Ilustrasi penggantian kata yang dilakukan terhadap kata-kata negatif didalam pesan yang dikirimkan pengguna dapat dilihat dibawah berikut:



Gambar 9 : Contoh tweet dengan beberapa kata negatif

Gambar 10 : Tweet dengan kata-kata negatif yang sudah diganti (*replace*)

Gambar 9 dan 10 mengambil contoh pengeposan status (tweet) pada jejaring sosial Twitter. Terlihat pada gambar 9, bahwa pengirim pesan ingin mengeposkan status dengan kata-kata negatif "bodoh", "ampas", "cacad" dan "mampus". Setelah tombol Tweet ditekan, maka *server* Twitter akan langsung mencari kata-kata negatif yang terdapat pada tweet yang ingin diposkan. Kata-kata tersebut kemudian dicocokan terhadap file eksternal yang telah didefinisikan sebelumnya. Setelah dilakukan pengecekan, ternyata ditemukan 4 kata negatif yang telah disebutkan diatas. Langkah berikutnya adalah mengganti kata-kata negatif tersebut dengan kata pengganti yang telah didefinisikan di file eksternal itu juga. Apabila terdapat banyak kata pengganti yang didefinisikan, maka akan dilakukan pemilihan *random* terhadap kata-kata pengganti yang akan digunakan. Akhirnya setelah kata-kata negatif tersebut berhasil diganti oleh kata-kata pengganti, maka Tweet akan berbunyi : "Farhat Abbas memang orang yang sangat pintar, oke dan bagus, Orang kaya gitu mending hidup saja."

Tidak ada lagi kata-kata negatif yang terkandung di dalam Tweet tersebut, dan makna nya dapat berubah drastis, seratus delapan puluh derajat dari Tweet awal yang penuh dengan kata-kata negatif. Ide yang sama dapat diaplikasikan kepada pengiriman pesan / *chatting* pada *game online.* Lalu bagaimana dengan kata-kata negatif yang didahului oleh kata tidak ? Pengecekan terhadap kata-kata negatif yang didahului oleh oleh kata tidak memiliki ide yang sama dengan ide penyensoran langsung terhadap kata-kata negatif pada poin A.

Meskipun dapat merubah makna secara signifikan atau mengalihkan makna sebelumnya yang negatif, tetapi tetap saja penggunaan kata-kata tertentu masih dapat menimbulkan terkaan terhadap makna negatif yang dimaksudkan oleh pengirim pesan.

Dari contoh gambar 9 dan 10, klausa "Orang kaya gitu mending hidup saja." masih dapat menimbulkan makna yang ambigu. Makna ambigu tersebut dapat berupa "Apakah memang pengirim pesan ingin menggunakan kata 'hidup' atau terjadi pergantian kata dari suatu kata menjadi kata hidup ?". Hal tersebut masih menjadi kekurangan dari ide pergantian langsung kata-kata negatif pada suatu pesan / tweet pada jejaring sosial / *game online.*

Konsep berikutnya, memberikan peringatan kepada pengirim pesan bahwa pesan yang ingin mereka kirimkan mengandung kata-kata negatif, dan pesan tersebut tidak ditampilkan ke layar penerima pesan. Konsep ini akan menutupi kekurangan yang ada pada ide pergantian langsung terhadap kata-kata negatif.

C. Pemberian Peringatan Langsung

Konsep sebelumnya, penggantian langsung kata-kata negatif yang terdapat pada pesan masih dapat menimbulkan makna ambigu terhadap pesan negatif yang sebenarnya dimaksudkan oleh pengirim. Oleh karena itu, tujuan akan ingin hilangnya kata-kata negatif (tervalidasi) pada pesan belum tercapai sepenuhnya.

Konsep pemberian peringatan langsung apabila ditemukan kata-kata negatif pada pesan akan mencegah semua bentuk kata-kata negatif yang dikirimkan oleh pengirim, sampai dan tampil pada layar penerima pesan.

Cara kerja dari konsep ini masih tergolong mirip dengan cara kerja kedua konsep sebelumnya, yaitu, ketika pengirim telah selesai mengetik pesan dan menekan tombol enter / send, *server game online* maupun jejaring sosial akan mengecek apakah didalam pesan tersebut masih terdapat kata-kata negatif atau tidak. Perbedaan konsep ini dengan kedua konsep sebelumnya adalah apabila ditemukan kata-kata negatif pada pesan dan kata-kata negatif tersebut terdapat pada file eksternal yang telah terdefinisi, maka akan ditampilkan peringatan bahwa terdapat kata-kata negatif pada pesan yang ingin dikirim kan, dan pesan tersebut dihapus, tidak ditampilkan pada layar penerima pesan. Apabila tidak ditemukan kata-kata negatif pada pesan sesuai dengan kata-kata negatif yang telah terdefinisi pada file eksternal, pesan tersebut akan ditampilkan pada layar penerima pesan.

Konsep ini dapat diterapkan baik di *game online* maupun jejaring sosial. Apabila ditemukan kata-kata negatif yang telah didahului oleh kata "tidak", maka sama dengan kedua konsep sebelumnya, pesan tersebut akan tetap dikirimkan, karena kata tidak akan menegasikan makna negatif pada pesan.

Intinya, penemuan satu buah kata negatif pada pesan yang akan dikirimkan yang tidak didahului dengan kata tidak dan telah didefinisikan pada file eksternal yang berisi kata-kata negatif akan men-*trigger* dikirimnya peringatan bahwa pesan yang

dikirimkan oleh pengirim, masih mengandung kata negatif. Kemudian pesan akan dihapus dan pengirim pesan diharuskan mengetik dan mengirim pesannya kembali, tentunya dengan memperhatikan kata-kata yang akan mereka pergunakan.

Ilustrasi pemberian peringatan yang dilakukan apabila ditemukan kata-kata negatif didalam pesan yang dikirimkan pengguna dapat dilihat dibawah berikut :

Pengirim : Woi asu ! kemana aja lu cacad ?
...
(tombol send ditekan)
...
SERVER : Pesan yang anda kirimkan mengandung kata kasar, negatif / berbau SARA
...
(Pengirim diharuskan mengetik pesan baru dan mengirimkannya kembali)

IV. KESIMPULAN

Pengaplikasian *String/Pattern Matching* dapat ditemukan dimana saja di kehidupan sehari-hari. Algoritma *String/ Pattern Matching* berguna untuk melakukan pencarian semua kemunculan *string* pendek (*pattern*) di dalam *string* yang lebih panjang (*teks*). Contoh dari algoritma *Pattern Matching* adalah algoritma *Brute Force*, algoritma Knuth-Morris-Pratt (KMP) dan algoritma Boyer-Moore. Masing-masing algoritma memiliki cara kerja, karakteristik, kompleksitas, keunggulan dan kekurangannya masing-masing.

Pattern Matching dapat diaplikasikan untuk memvalidasi pesan yang akan dikirimkan oleh pengirim ke penerima pada media sosial seperti *game online* dan situs jejaring sosial. Ide-ide atau konsep-konsep yang dapat diterapkan sebagai bentuk validasi terhadap pesan yang mengandung kata-kata negatif adalah :

1. Menyensor langsung kata-kata negatif di dalam pesan.
2. Mengganti kata-kata negatif di dalam pesan menjadi kata-kata positif yang merupakan lawan kata dari kata-kata negatif atau kata-kata plesetan dari kata-kata negatif tersebut.
3. Memberikan peringatan kepada pengguna yang mengirimkan pesan dengan kata-kata negatif bahwa pesan yang ingin dikirim mengandung kata-kata yang negatif atau kurang baik.

Dalam merealisasikan ketiga konsep tersebut, dibuat sebuah file eksternal yang berisikan kata-kata negatif yang mungkin diketikkan oleh pengirim pesan. Khusus untuk konsep nomor 2, ditambahkan kata-kata pengganti untuk setiap kata-kata negatif. Kata-kata pengganti ini yang akan ditampilkan pada layar pengirim dan penerima pesan. Algoritma yang digunakan sebagai algoritma *pattern matching* adalah algoritma Boyer-Moore, karena algoritma ini akan berjalan dengan cepat apabila pesan yang dikirimkan adalah dalam bentuk alfabet dan numerik.

Konsep pertama akan mengganti kata-kata negatif dengan karakter bintang (*). Konsep kedua akan mengganti kata-kata negatif yagn terdapat di dalam pesan dengan kata-kata pengganti yang telah didefinisikan sebelumnya di file eksternal. Konsep ketiga akan memunculkan peringatan apabila pesan yang dikirimkan mengandung kata-kata negatif yang terdefinisi di dalam file eksternal. Jika kata-kata negatif tersebut didahului oleh kata tidak, maka pesan tersebut tidak perlu divalidasi, langsung ditampilkan saja pada layar pengirim dan penerima pesan.

DAFTAR REFERENSI

[1] Kaplan, Andreas M.; Michael Haenlein (2010) "*Users of the world, unite! The challenges and opportunities of Social Media*". Business Horizons 53(1): halaman 59–68
Waktu akses : 18 Desember 2013, 19.30 WIB

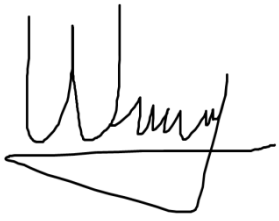
[2] Slide Presentasi Pencocokan String, Bahan kuliah IF2211 Strategi Algoritma, Teknik Informatika ITB

[3] Munir, Rinaldi, 2009, *Diktat Kuliah IF2211 Strategi Algoritma*, Bandung, Informatika Bandung.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 20 Desember 2013



Wishnu
13511040